Unleashing software developers

May 2013
A call to action

• 25 years is too long to wait for
  – “Reasonable” FPGA development tools
  – “Powerful” FPGA development tools
  – Tools that enable reconfiguration and highly parallel designs usable by all developers
Convergence
Operational Technology

Operational Technology: Capture data in real time

Unstructured data is so voluminous and fast it requires eHPC computing at the edge
Information Technology

Informational Technology:
Turn that data into useful information

Complex data sets require HPC for information development
Business Technology

Business Technology:
Turn that information into reduced cost

Use HPC because - Time is Money
Business Technology:
Turn that information into less risk

Use HPC because - Knowledge is Power
Business Technology:
Turn that information into revenue

Use HPC because – Revenue drives everything
Inflection Point: Parallel Needs Across Markets
Pretty Pictures, Nice Story, But…

The real world comes with real problems

There is just way too much data – to collect
A world of scale

• Billions
  – People
  – Internet of things
  – IC devices
• Peta-scale computing
• Peta-scale storage
• Sensors collecting in real-time a data flood
Reconfigurable computing is missing

- From the real world
- From real problems and applications
- But the real world needs what it enables
  - Powerful computation
  - Efficiency
  - Robustness
  - Innovation
Today’s barriers

• Tools
• Device capabilities
• Accessibility
• Software engineering role
• Compute models
Today’s Reconfigurable tools are too hard

- Fragmented into many tools
- File formats differ
- Binaries differ
- Hardware centric
- Lack cohesive computing model
- Difficult to discover
- Learn and use
- Incomplete
SW Developers to unleash

• More SW developers then HW developers
  – More then 100:1 difference
• SW drives most real world applications
  – Platforms
  – Adoption rates
  – Utility
  – Retention of users
• SW applications dominate
  – Think App stores
INSERT WORD

2A1  BANANAS
2A4  CARROTS
2A5  LETTUCE
2A6  BEANS
2B  CANS
2C  CEREALS
2D  COLD LOCKER
2E  FROZEN LOCKER
2F  MISCELLANEOUS
3  SHOE STORE
4  HARDWARE
5  ART SUPPLY
6  DRUG STORE
7  LIBRARY
Doug Engelbart

• SRI demo 1968
  – Interactive computing
  – Distributed computing
  – Real-time editing
  – Cut and paste
  – Mouse and chord to extend keyboard
  – Live multi-city video conferencing
  – Broadband
Andrew Singer

• THINK Pascal - 1986
  – First tool for Mac developers
  – Integrated Development Environment
  – Instantaneous compile (< 30 seconds)
  – Automated MAKE system
  – Debugging integrated with Development
    • Active debugging, trace, stop points, dynamic variables
  – Copied by all following GUI SW platforms
    • Apple, Microsoft, SUN, Eclipse, Google
What is powerful

• Accelerate the ability to move from idea to implementation
  – Example Internet startup versus Hardware startup

• Allow all designs to scale across multiple devices and vendor devices

• Enable the developer to focus on the design not the implementation

• Enable true re-use (module, core, and app)
What is reasonable

• Tools that remove the burden from the developer of endless details
  – Timing closure
  – Area planning
  – Speed grade choice
• Fast compile
  – PAR in minutes
• Enable reconfiguration
Real world design

- Algorithm tools
  - MathWorks, Mathematica, Simulink, paper, spreadsheet, or drawing

- Hardware tools
- Software tools
- Complex environment
- Difficult verification
- Changes difficult
- Time Consuming
- Design limiting
Seeking One design environment

- HW and SW Unified flow
- Common code base
- Migrate from HW to SW with ease
- Migrate across device types with ease
- Reuse across device generations
- Unified verification and test
- Enable accelerate design discovery and implementation
Natural computing

• Emulates the biological world
• Change is embraced and used
• Diverse set of compute elements
• Simple and repeatable across system
• Scales from one to million
• Design reuse
Support Reconfigurable Categories

• Dynamic Reconfiguration
• Active Dynamic Reconfiguration
• Cooperating Dynamic Reconfiguration
• Evolvable Reconfiguration
• Cognitively Evolvable Reconfiguration
Dynamic Reconfiguration

• Runtime Dynamic
  – Time and space domains

• Function level
  – Data driven
  – Event driven
  – Function level granularity

• High burden on designer/architect
  – Limited real world use without effective tools
Active Dynamic

• Across device boundaries
• Higher then function level
  – Modules, libraries, units
  – Can re-locate across devices
• Across system boundaries
  – Internet of Things
• Multiple actors initiate change
  – Events, time, data, device
• Enables robustness
  – Health monitors relocate prior to failure
Cooperating Dynamic

• Application level
  – Changes as Application sets change
  – Change as System environments change

• User driven actions
  – Initiate changes
  – Anticipate changes
  – Adapt to User patterns and modes
Evolvable

• Building blocks
  – Regular, repeating, simple, extendable

• Mutating
  – Introduces random change, chance and environmental activated

• Competing changes

• Survival of fittest
  – Evolve at design stage
  – Evolve in-situ
Cognitively Evolvable

• AI or Machine learning coupled
• Evolutionary monitor
  – Mange in-situ changes
  – Learning
  – Threshold monitors
  – Fitness monitors
  – System monitors
Summary

• Make this leap to enable SW developers
• Make this leap to enable all these modes of reconfigurable
• Then Reconfigurable moves into the mainstream
• And then we will see something that will amaze us all